Attorney Docket No.: 017900-004110US
Client Reference No.: 2003E00651 IL

PATENT APPLICATION

TECHNIQUES FOR PREVIEWING CONTENT PACKAGE FILES THROUGH A PORTAL

Inventor: Oded Grinberg, a citizen of Israel, residing at
11/19 Got Levin St.
Haifa, 32922 Israel

Assignee: SAP AG
Neurottstrasse 16
D-69190 Walldorf
Germany

Entity: Large

TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, 8th Floor
San Francisco, California 94111-3834
Tel: 650-326-2400

TECHNIQUES FOR PREVIEWING CONTENT PACKAGE FILES THROUGH A PORTAL

CROSS-REFERENCES TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Application No. 60/519,007, filed November 10, 2003, which is incorporated by reference herein.

BACKGROUND OF THE INVENTION

[0002] The present invention relates to techniques for previewing content package files through a portal, and more particularly, to techniques for previewing content package files through a portal interface before importing the content package files into the portal.

[0003] A content package file is a relational data structure file that contains references to other content files. The other content files reside in the same directory as the content package file or in subdirectories of the package content file directory. A content package file may be written in an Extensible Markup Language (XML), for example.

[0004] Client-server systems can import content package files to a portal server computer. When a content package file is imported, all of the content files referenced in the content package file are also imported to the portal server computer. The user can view the content package file using a portal.

[0005] When a content package file is imported, all of the other content files referred to in the content package file are also imported and stored on the portal server computer. One or more of the referenced content files may be duplicative of a content file currently stored on the portal server computer.

[0006] When the referenced content files are imported and "update" mode is used, these referenced content files automatically overwrite any content files previously stored on the portal server computer that have the same names. If the user has, in the interim, updated or otherwise edited content files stored on the portal server computer, those updates and edits are lost if a content file with the same name is imported. This creates a problem, because users have no reliable way of knowing which other content files are referenced in the package

content file, until after the package content file and the files it references are imported, and previously updated files with the same name have been erased.

[0007] Other problems occur when content package files are imported to a portal server computer. As discussed above, content package files contain references to other content files in the form of path names. These path names in the content package file may contain errors. For example, the path name may be misspelled. In this case, the misspelled path name does not refer to a valid path. As another example, a path name in the content package file may not refer to path where a content file is actually stored. This type of error can occur if a content file is moved from one location to another.

[0008] When a user imports a content package file that contains an incorrect path name, the content file referenced by the incorrect path name is not imported to the portal server computer. Instead, the user must locate and then import this content file separately.

[0009] It would therefore be desirable to provide techniques for importing content package files through a portal that eliminate these problems.

BRIEF SUMMARY OF THE INVENTION

[0010] The present invention provides techniques for previewing a content package file through a portal interface before the file is imported to the portal server computer. When a user requests to preview a content package file, the present invention analyzes the content package file and extracts basic information from the content package file.

[0011] The content package file contains references to a first level of other content files. The first level content files within the content package file can contain references to second level content files. The second level content files can contain references to third level content files, and so on.

[0012] The content from the content package file and the other contents files can be packaged into one large file. Basic information is extracted from this large file that summarizes the content files and their relationships to each other and the content package file. The basic information is then displayed in a preview screen to the user through a portal interface before the content files are imported.

[0013] A user can be notified if any of the references to content files contain errors. This feature allows a user to correct an erroneous reference before the content files are imported onto the portal server computer.

[0014] Other objects, features, and advantages of the present invention will become apparent upon consideration of the following detailed description and the accompanying drawings, in which like reference designations represent like features throughout the figures.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] Figure 1 illustrates a process for allowing a user to preview a content package file and the content files referred to within the content package file according to an embodiment of the present invention;

[0016] Figure 2 illustrates a client-server system diagram that implement an embodiment of the present invention;

[0017] Figure 3 illustrates an example of a display screen that allows a user to select a content package file to preview according to the present invention; and

[0018] Figures 4A-4B illustrate examples of how a content package file and its sub-files can be previewed on a displayed screen in an expandable hierarchical format according to the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0019] The present invention allows a user to preview a content package file through a portal before the file is imported to a portal server. Figure 1 is a flow chart that illustrates a process according to an embodiment of the present invention.

[0020] Figure 2 illustrates an example of a client-server system diagram that can implement an embodiment of the present invention. The system diagram of Figure 2 includes a client machine 150 and a portal server machine 151. The process described in Figure 1 can be implemented by the client-server system shown in Figure 2.

[0021] Referring to Figure 1, initially, a user is allowed to select a content package file at step 110. Figure 3 illustrates an example of a display screen 210 that can be provided to a user to allow the user to select a content package file at step 110. A user can type the path to

a content package file directly in region 211 or browse through directories of files to locate the path to a content package file.

[0022] Screen 210 allows a user to choose whether to import a file from a client or a server machine. The user can import a content package file from a server machine or from a client machine.

[0023] A user can select preview option 212 in screen 210 to preview the content package file specified in region 211 according to the principles of the present invention. A user can select import option 213 to import the content package file specified in region 211.

[0024] When a user selects preview option 212, the content package file specified in region 211 is sent to server 151 at step 111 as shown in Figures 1 and 2. Server 151 extracts the references to the content files referred to within the content package file. Server 151 then fetches the content files at step 112.

[0025] Content files referred to in the content package file are called first level content files. Content files referred to in a first level content file are called second level content files. Content files referred to in a second level content file are called third level content file, etc.

[0026] Portal server 151 iterates through each of the content files, extracts from them the references to other content files (i.e., the second level, third level, etc. content files), and then fetches these other content files from client 150 at step 112. This iterative process stops when all of the references to content files have been extracted and fetched.

[0027] The portal server 151 establishes communications with client 150 to transfer the content files to portal server 151 at step 112. These communications are depicted in Figure 2. In an alternative embodiment of the present invention, the content files are transferred to portal server 151 from another server machine.

[0028] At step 113, the contents of the content files fetched at step 112 are extracted. All of the content from the content file is extracted including all of its data and code. Also at step 113, all of the references to the content files are replaced with the extracted content. A large combined file (e.g., a large XML file) is created at step 113. In this combined file, all of the references are eliminated from the content package file and the content files.

[0029] At step 114, a preview screen of the content package file is created using the combined file generated at step 113. The portal server renders basic information about the

content files from the combined file to create the preview screen. Portal server 151 then sends this preview display to client machine 150.

[0030] In general, at step 114, the basic information rendered about the content files is less than all of the content in the content files. For example, the basic information about the content files can include meta-data such as names of the files, file types, version numbers, etc. The basic information can also include brief summaries of the content files.

[0031] The preview can be an HTML file format, or any other suitable file format used for displaying information. In one embodiment of the present invention, portal server 151 renders an XML combined file to create a preview HTML display.

[0032] After the server creates the preview screen at step 114, server 151 sends the preview screen to client 150 to be displayed on the client machine. At step 115, the preview screen of the content package file is displayed. The preview can display the basic information about the content files and the content package file. In one embodiment of the present invention, the preview screen displays an expandable hierarchy of the content package file and the referenced content files.

[0033] Steps 112-114 are typically performed on portal server 151. Portal server 151 establishes a communication with a client machine to fetch each of the content files. This technique requires the portal server to establish multiple communications with the client to obtain the content files.

[0034] To reduce the number of communications a portal server makes with the client, a user can compress the content package file and its referenced content files into a compressed file format, such as, for example, a Zip file. According to this embodiment, the portal server gets the compressed file. The server can then expand the file contents and generate the preview of the content package file. By compressing the content package file, the number of communications that the server has to make to fetch each of the content files is reduced. Therefore, the portal server can create the preview screen in less time.

[0035] Figures 4A-4B are examples of preview screens that preview a content package file according to the present invention. The preview screens shown in Figures 4A-4B display a preview of a content package file in an expandable hierarchy.

[0036] According to this example, a user can click on the + (plus) or − (minus) icon next to the package to preview first level content files that are referred to in the content package file.

A plus icon indicates that a content file contains references to other content files. A user can click on the plus icon to expand the next level of the hierarchy in order to view the referenced content files. When a minus icon is displayed next to a file in the preview screen, either there are no lower level content files referenced in that content file or the lower level content files are already displayed. It should be clear from this description that other icons may be used to indicate this information.

[0037] In the example of Figures 4A-4B, the preview screen displays a file name, a file type, and a version number for the content file named "myiView." For the content file name "myPage," the preview displays a file name and a Boolean value (contentUrl) indicating whether the myPage file refers to other content files. First level content file labels for myiView and myPage are displayed beneath the package content file name in an hierarchical format that is expandable. The user can click on the minus icon next to "package" to hide the first level content files as well as the other levels of content files.

[0038] The user can click on the plus icon next to content file name myPage in Figure 4A to preview the second level content files that are referenced in the myPage file as shown in Figure 4B. In this example, the myiView content file does not contain references to any other content files, and as a result, the myiView node cannot be expanded.

[0039] When the display for the content file myPage is expanded, the second level content files that are referenced in myPage are previewed beneath the listing of dependencies. For example, the content file iViewPage is referenced in the myPage file, so iViewPage is displayed in the dependencies list beneath the myPage file label in the hierarchical format.

[0040] If the iViewPage file contains references to other third level content files, then a plus icon is displayed next to the file label. By clicking on this plus icon, the iViewPage node can be expanded to preview the third level content files. In the example of Figure 4B, the iView Page file does not contain any references to other content files, so a minus icon is displayed next to its label.

[0041] The present invention displays basic information next to each content file label in the preview screen. In the examples of Figures 4A-4B, file names, file types, and version numbers are displayed as part of the preview of the content files. In addition to this information, the preview can display a brief summary of the contents of one or more of the content files, or possibly some or all of the contents of the file.

[0042] According to the present invention, any data or code from the content files can be displayed in the preview screen through a portal. The purpose of the preview screen is to let the user know which content files are embedded in a file hierarchy beneath the main content package file. Additional basic information about the file such as a file type, name and a brief summary of its content can be displayed to help the user better understand the nature of each of the embedded content files.

[0043] Based on the information displayed about each content file in the preview screen, the user can decide whether to import the content package file. If the user decides to import the content package file, the user clicks on import option 213. The package content file and all of the lower level embedded content files are then imported to the portal server computer at step 116.

[0044] The present invention also allows a user to determine whether the content files imported to the portal server computer overwrite existing content files with the same name. Drop down list 214 in the display screen of Figure 3 gives the user the option to select whether the imported content files overwrite existing content files. In Figure 3, the user has chosen 'none' from the list, which prevents the imported content files from overwriting any of the existing content files stored on the portal server computer. This feature allows a user to prevent an imported content file from overwriting an existing file that has been updated or otherwise changed with respect to the imported file.

[0045] Because the present invention displays each of the lower level content files in the preview screen before the content files are imported, the user can see in the preview screen whether any of the content files may overwrite his existing files. If such overwrite files exist, then the user selects the 'none' option from list 214 to prevent his existing files from being lost.

[0046] If the user wants content files displayed in the preview screen to overwrite existing files, the user can select a different option from list 214. Drop down list 214 has an option that causes imported files to overwrite any existing files with the same name on the portal server computer.

[0047] As discussed above, one (or more) of the references to the content files may contain an error. There are two types of errors. One type of error occurs when the wrong path name is entered as the reference. Another type of error occurs when the correct path name is

initially entered as the reference, but the content file is subsequently moved to another location.

[0048] In any event, when one of the content file references is erroneous, the system detects the error when it tries to fetch the content file at step 112. The preview screen then displays an error message to the user indicating that the reference to the file contains an error. The user can then fix the erroneous reference and then attempt to preview the content files again.

[0049] This feature is advantageous, because it allows the user to fix any errors in the content file references before the content package file is imported to the portal server computer. This provides the user with enhanced control over the content of the files that are imported onto to the portal server computer. Without this feature, content files linked to erroneous references are not imported to the portal server computer.

[0050] While the present invention has been described herein with reference to particular embodiments thereof, a latitude of modification, various changes, and substitutions are intended in the present invention. In some instances, features of the invention can be employed without a corresponding use of other features, without departing from the scope of the invention as set forth. Therefore, many modifications may be made to adapt a particular configuration or method disclosed, without departing from the essential scope and spirit of the present invention. It is intended that the invention not be limited to the particular embodiments disclosed, but that the invention will include all embodiments and equivalents falling within the scope of the claims.